Innovations Syst Softw Eng (2005) 1: 79–88
DOI 10.1007/s11334-005-0001-5

STATE OF THE ART

Roy Sterritt

Autonomic computing

Received: 1 December 2004 / Accepted: 17 January 2005 / Published online: 11 March 2005

Abstract Autonomic computing (AC) has as its vision the creation of self-managing systems to address today's concerns of complexity and total cost of ownership while meeting tomorrow's needs for pervasive and ubiquitous computation and communication. This paper reports on the latest autonomic systems research and technologies to influence the industry; it looks behind AC, summarising what it is, the current state-of-the-art research, related work and initiatives, highlights research and technology transfer issues and concludes with further and recommended reading.

1 Introduction

Autonomic computing (AC), as the name suggests, is a metaphor based on biology. The autonomic nervous system within the body is central to a substantial amount of nonconscious activity that allows us as individuals to proceed with higher level activity in our daily lives [1]. Typical examples that have been highlighted are heartbeat rate, breathing rate, reflex reactions upon touching a sharp or hot object and so on [2–4]. The aim of using this metaphor is to express the vision to enable something similar to be achieved in computing, in other words, to create the self-management of a substantial amount of computing function to relieve users of low-level management activities, allowing them to place emphases on the higher level concerns of running their business, their experiments or their entertainment.

The need and justification for AC is based on the ever increasing complexity in today's systems. It has been expressed that the information technology (IT) industry's single focus has been on improving hardware performance, with software burgeoning with additional features to maximise on this additional capacity, at the neglect of other vital criteria. This has created a trillion-dollar industry with consumers consenting to the hardware–software upgrade cycle. Its legacy, though, is a mass of complexity within systems of systems, resulting in an increasing financial burden per computer (often measured as the TCO: total cost of ownership).

In addition to the TCO implications of complexity, complexity in itself is a blocking force to achieving dependability [5]. Dependability, a long-standing desirable property of all computer-based systems, integrates such attributes as reliability, availability, safety, security, survivability and maintainability [6]. Dependability was identified by both US and UK Computer Science Grand Research Challenges; "Build systems you can count on", "Conquer system complexity" [7] and "Dependable systems (build and evolution)" [8]. The autonomic initiatives offer a means to achieve dependability while coping with complexity [5].

Initial reaction to the autonomic initiative was "is there anything new?", and to some extent, this question can be justified as artificial intelligence (AI) and fault tolerant computing (FTC), among other research disciplines, have been researching many of the envisaged issues within AC for many years. For instance, the desire for automation and effective robust systems is not new; in fact, this may be considered an aspect of best practice systems and software engineering. Similarly, the desires for systems self-awareness, awareness of the external environment and the ability to adapt are also not new, being major goals of several fields within AI research. What is new is AC's holistic aim of bringing all the relevant areas together to create a change in the industry's direction; selfware instead of the hardware and software feature upgrade cycle of the past that created the complexity and TCO quagmire. Yet, a danger lies in that the self-* properties may just become the next marketing cycle of features–autonomics inside, and ultimately, AC would fall prey to the public impression of not meeting perceived expectations. As such, it must be kept in the foreground that this is a long-term strategic initiative with evolutionary deliverables enroute.

R. Sterritt
School of Computing and Mathematics,
Faculty of Engineering,
University of Ulster,
Jordanstown Campus,
Newtownabbey County Antrim,
Northern Ireland BT37 0QB
E-mail: r.sterritt@ulster.ac.uk

This paper is part of the NASA ISSE Journal's State-of-the-Art series reporting on the latest research and technologies to influence the industry; it looks behind AC, summarising what it is, the current state-of-the-art research, related work and initiatives and highlights research and technology transfer issues and concludes with further and recommended reading.

2 What is autonomic computing?

IBM, upon launching the call to the industry, voiced the state of the industry's concerns as complexity and TCO. They presented the solution to be AC, expressed as comprising eight elements [1];

- Possess system identity—detailed knowledge of components
- Self-configure and reconfigure—adaptive algorithms
- Optimise operations—adaptive algorithms
- Recover—no impact on data or delay on processing
- Self-protection
- Be aware of environment and adapt
- Function in a heterogeneous world
- Hide complexity

These eight elements can be expressed in terms of properties that a system should have to constitute autonomicity [5]. These are described in Sect. 2.1 and elaborated upon in Sect. 2.2, which discusses the very constructs that constitute these properties.

2.1 Autonomic properties

The properties that a system should have to constitute autonomicity are depicted in Fig. 1 [5].

The general properties of an autonomic (self-managing) system can be summarised as four objectives: self-configuring, self-healing, self-optimising and self-protecting; and four attributes: self-awareness, environment-awareness, self-monitoring and self-adjusting (Fig. 1). Essentially, the objectives represent broad system requirements while the attributes identify basic implementation mechanisms. Since the 2001 launch of AC, the self-* list of properties has grown substantially [9] (refer to Sect. 6. Defining terms, terminology and glossary); yet this initial set still represents the general goal.

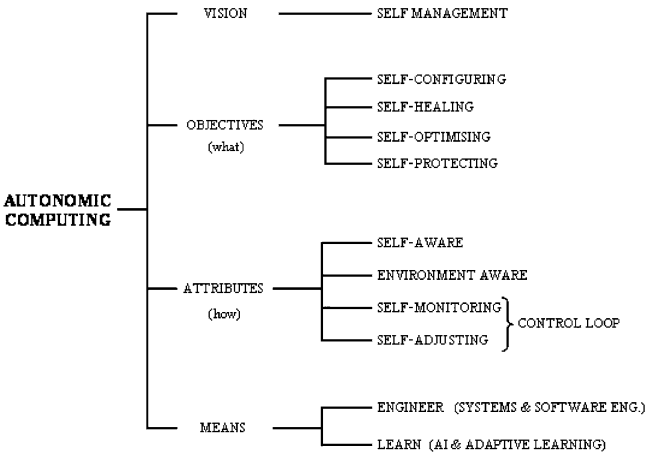

Fig. 1 Autonomic computing properties tree

Self-configuring is a system's ability to readjust itself automatically; this may simply be in support of changing circumstances or to assist in self-healing, self-optimisation or self-protection. *Self-healing*, in reactive mode, is a mechanism concerned with ensuring effective recovery when a fault occurs; identifying the fault and then, where possible, repairing it. In proactive mode, it monitors vital signs in an attempt to predict and avoid health problems. *Self-optimisation* means that a system is aware of its ideal performance, can measure its current performance against that ideal and has policies for attempting improvements. It may also react to policy changes within the system as indicated by the users. A *self-protecting* system will defend itself from accidental or malicious external attack. This means being aware of potential threats and ways of handling those threats (Fig. 1) [5].

In achieving such self-managing objectives (Fig. 1), a system must be aware of its internal state (self-aware) and current external operating conditions (environment-aware). Changing circumstances are detected through self-monitoring and adaptations are made accordingly (self-adjusting) [5]. As such, a system must have knowledge of its available resources, its components, their desired performance characteristics, their current status, and the status of interconnections with other systems, along with rules and policies of how these may be adjusted. Such ability to operate in a heterogeneous environment will require the use of open standards to enable global understanding and communication with other systems [1].

These mechanisms are not independent entities; for instance, if an attack is successful, this will include self-healing actions and a mix of self-configuration and self-optimisation; in the first instance, to ensure dependability and continued operation of the system and, later, to increase the self-protection against similar future attacks. Finally, these self-mechanisms should ensure there is minimal disruption to users, avoiding significant delays in processing.

There are two main perceived approaches (Fig. 1) considered to be the means for AC to become a reality [4]:

- Engineer autonomicity
- Learn autonomicity

Engineer autonomicity has an implied systems and/or software engineering view; to engineer autonomic function into the individual systems.

Learn autonomicity has an implied AI, evolutionary computing and adaptive learning view; to utilize algorithms and processes to achieve autonomic behaviour.

However, both approaches rely on each other to be able to achieve the objectives set out in AC. As such, AC may prove

to require a greater collaboration between the intelligence systems research and system and software engineering fields to achieve the envisaged level of adaptation and self-management within the autonomic initiative.

2.2 Necessary constructs

To meet these autonomic properties, the key constructs and principles that constitute an autonomic environment are

- Selfware; self-*
- AE = MC + AM
- Control loop; sensors + effectors
- AE⇔AE

Selfware*; Self-**.

The principle of selfware (self-managing software/firmware) and the need for self-* properties were discussed in the previous sections.

AE=MC+AM.

Figure 2 represents a view of an architecture for an autonomic element that consists of the component requiring to be managed and the autonomic manager [10, 11]. It is assumed that an autonomic manager (AM) is responsible for a managed component (MC) within a self-contained autonomic element (AE). This autonomic manager may be designed as part of the component or provided externally to the component, for instance, as an agent. Interaction will occur with remote autonomic managers (ref Fig. 2's autonomic communications channel) through virtual, peer-to-peer, client-server [12] or grid [13] configurations.

Control loop; sensors + effectors

At the heart of any autonomic system architecture are sensors and effectors [2]. A control loop is created by monitoring behaviour through sensors, comparing this with expectations (knowledge, as in historical and current data, rules and beliefs), planning what action is necessary (if any) and then executing that action through effectors [14]. The control loop (full name; closed loop of feedback control), a success of manufacturing science for many years, provides the basic backbone structure for each system component [15].

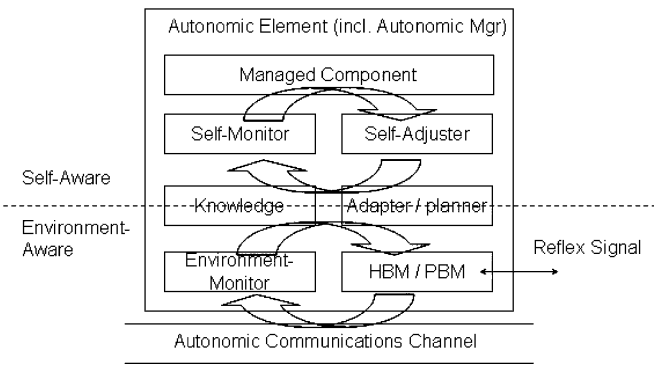

Fig. 2 Autonomic element consisting of autonomic manager and managed component

IBM represents this self-monitor–self-adjuster control loop as the monitor, analyse, plan and execute (MAPE) control loop. The monitor-and-analyse parts of the structure process information from the sensors to provide both self-awareness and an awareness of the external environment. The plan-and-execute parts decide on the necessary self-management behaviour that will be executed through the effectors. The MAPE components use the correlations, rules, beliefs, expectations, histories and other information known to the autonomic element or available to it through the knowledge repository within the AM.

AE ⇔ AE.

The autonomic environment requires that autonomic elements and, in particular, autonomic managers communicate with one another concerning self-* activities to ensure the robustness of the environment. Figure 2 views an AE with the additional concept of a pulse monitor—PBM (an extension of the embedded system's heart-beat monitor (HBM), which safeguards vital processes through the emission of a regular 'I am alive' signal to another process) with the capability to encode health and urgency signals as a pulse [16]. Together with the standard event messages on the autonomic communications channel, this provides dynamics within autonomic responses and multiple loops of control, such as reflex reactions among the autonomic managers [17].

2.3 Evolution versus revolution

Due to the need for the differing levels of human involvement and facing the reality that the overarching vision of AC cannot be achieved overnight, AC maturity and sophistication has been categorized into five stages of adoption [11, 18, 19]: basic, managed, predictive, adaptive and autonomic.

Assessing where a system resides within these autonomic maturity levels is not necessarily an easy task. Efforts are underway to define the characteristics and metrics that are required to be measured [20]. The overall AC maturity is established from a combination of dimensions forming a natural continuum of autonomic evolution [21], such as increasing functionality (manual, instrument and monitor, analysis, closed loop to closed loop with business priorities) and increasing scope (subcomponents; single instances; multiple instances, same type; multiple instances, different types; to business systems) [21]. Because assessment is becoming even more complex, efforts are currently underway to automate the assessment process itself [22].

These efforts imply that the autonomic initiative is an evolutionary path to achieve a revolution.

3 State-of-the-art research

In the embryonic years of AC, there was a substantial amount of research effort in the area. The reason the initiative hit the ground running is that many disparate research fields paved the way.

It has been highlighted that meeting the grand challenge of AC systems will involve researchers in a diverse array of fields, including systems management, distributed computing, networking, operations research, software development, storage, artificial intelligence and control theory, as well as others [2]. There is not the space in this paper to cover all the excellent research underway, yet this section will discuss a selection of the early reports in the literature of state-of-the-art efforts in AC [23].

3.1 Machine design

An interesting paper in [24] discusses affect and machine design [25]. Essentially, it supports those psychologists and AI researchers that hold the view that affect (and emotion) is essential for intelligent behaviour [26,27]. It proposes three levels for the design of systems:

1. Reaction—lowest level, where no learning occurs but immediate response to state information coming from sensory systems.
2. Routine—middle level, where largely routine evaluation and planning behaviours take place. It receives input from sensors as well as from the reaction level and reflection level. This level of assessment results in three dimensions of affect and emotion values: positive affect, negative affect and (energetic) arousal.
3. Reflection—top level receives no sensory input or has no motor output, it receives input from below. Reflection, a meta-process, where the mind deliberates about itself. Essentially, operations at this level look at the systems representations of its experiences, its current behaviour, its current environment etc.

Essentially, the reaction level sits within the engineering domain, monitoring the current state of both the machine and its environment, with rapid reaction to changing circumstances. The reflection level may reside within the AI domain utilising its techniques to consider the behaviour of the system and learn new strategies. The routine level may be a cooperative mixture of both.

3.2 Prediction and optimisation

Clockwork, a method for providing predictive autonomicity by regulating its behaviour in anticipation of need, using statistical modelling, tracking and forecasting methods [28], is now being expanded to include real-time model-selection techniques to fulfil the self-configuration element of AC [29]. This work contains probabilistic reasoning and could benefit from an inclusion of utilising a genetic algorithm for the model selection.

The use of probabilistic techniques such as Bayesian networks (BNs), discussed in [30], are also central to the research of autonomic algorithm selection. The system uses the BN approach along with self-training and self-optimising to find the best algorithm [30].

In effect, the breadth and scope of the autonomic vision is highlighted by such works that use AI techniques (machine learning, Tabu search, statistical reasoning and clustering analysis) for controlling the detection of the need for reoptimisation of enterprise business objectives [31].

An example of a ubiquitous computing application—smart doorplates— seeks to assist visitors to a building in locating an individual who is presently not in their office. A module in the architecture utilises probabilistic reasoning to predict the next location of the individual, which is reported along with their current location [32,33].

3.3 Knowledge capture and representation

A vital issue to the success of AC is the ability to transfer knowledge about the system management and configuration from human experts to the software managing the system. Fundamentally, this is a knowledge-acquisition problem [34]. One current research approach is to automatically capture the expert's actions (keyboard and mouse movements etc.) when performing on a live system and dynamically build a procedure model that can execute on a new system to repeat the same task [34]. Establishing a collection of traces over time should allow the approach to develop a generic and adaptive model.

The Tivoli management environment approaches this problem by capturing the key characteristics of a managed resource in its resource model [35]. This approach is being extended to capture the best practices information into the common information model (CIM) through descriptive logics at both the design phase and the deployment phase of the development life cycle [36]. In effect, the approach captures system knowledge from the creators, ultimately to perform automated reasoning when managing the system.

3.4 Monitoring and root-cause analysis

Event correlation, rule development and root-cause analysis are important functions of the autonomic environment [37]. Early versions of tools or autonomic functionality updates to existing tools and software suites in this area have recently been released by IBM [15] through their AlphaWorks Autonomic Zone website [22].

The generic log and trace tool correlates event logs from legacy systems to identify patterns. These can be used to facilitate automation or help debugging [15].

The Tivoli autonomic monitoring engine essentially provides server-level correlation of multiple IT systems to assist with root cause analysis and automated corrective action [15].

The ABLE rules engine can be used for more complex analysis. In effect, it is an agent-building learning environment that includes time series analysis and Bayes classification, among others. It correlates events and invokes the necessary action policy [15].

It has been highlighted that correlation, rule discovery and root-cause analysis activity can benefit from incorporating Bayesian networks [38] either in the rule-discovery process or in the actual model learning to assist with self-healing [39].

Large-scale server management and control have also received similar treatment. Event logs from a 250-node large-scale server were analysed through applying a number of machine-learning algorithms and AI techniques to establish time-series methods, rule-based classification and Bayesian network algorithms for a self-management and control system [40].

Another current aspect is the calculation of costs in an autonomic system and the self-healing equation. One approach utilises naive Bayes for cost-sensitive classification and a feedback approach based on a Markov decision process for failure remediation [41]. The argument is easily made that the autonomic system involves decisions and that decisions involve costs [42]. This naturally leads to work with agents, incentives, costs and competition for resource allocation and extensions thereof [42,43].

3.5 Legacy systems and autonomic environments

Autonomic computing is widely believed to be a promising approach to developing new systems. Yet, organisations continue to have to deal with the reality of legacy systems or build systems of systems comprising new and legacy components involving disparate technologies from numerous vendors [44]. Work is currently underway to add autonomic capabilities to legacy systems in areas such as instant messaging, spam detection, load balancing and middleware [44].

Generally, the engineering of autonomic capability into legacy systems involves providing an environment that monitors the sensors to the system and provides adjustment through effectors to create a control loop.

One such infrastructure is KX (Kinesthetics eXtreme), which runs a Lightweight, decentralized, easily integrated collection of active middleware components tied together via a publish-subscribe (content-based messaging) event system [44]. The Astrolabe tool may be used to automate self-configuration and monitoring and to control adaptation [45].

The AutoMate project, incorporating ACCORD, an autonomic component Framework, utilises DIOS to provide mechanisms to directly enhance traditional computational objects/components with sensors, actuators, rules, a control network, management of distributed sensors and actuators, interrogation, monitoring and manipulation of components at runtime through a distributed-rule engine [46].

3.6 Space systems

Increasing constraints on resources and greater focus on the cost of operations have led NASA and others to utilise adaptive operations and move toward almost total onboard autonomy in certain classes of mission operations [47,48]. Autonomy provides self-governance, giving responsibility to the agents within the system to meet their defined goals. Autonomicity provides self-management in addition to self-governance to meet one's own functional goals. There is also a shared responsibility to ensure the effective management (through self-* properties) of the system, which may include responsibilities beyond the normal task-oriented goals of an individual agent; for instance, the monitoring of another agent's health signs (ensuring self-protection, self-healing and if any issues initiate self-configuration and/or self-optimisation activities). As such, AC has been identified by NASA as a key area [4, 49–51] and research is underway to utilise it in addition to autonomy [52].

3.7 Agents

Agents, with their autonomous ability, have the potential to play a major role in AC [1,9,43,53–55]. At this stage, there are no assumptions that agents have to be used in an autonomic architecture but, as in complex systems, there are arguments for designing the system with agents [56], as well as providing inbuilt redundancy and greater robustness [57], through to retrofitting legacy systems with autonomic capabilities that may benefit from an agent approach [44]. With reference to work previously mentioned, a potential contribution of agents may come from learning environments, for the rules and norms or agent-monitoring systems.

3.8 Policy-based management

Policy-based management becomes particularly important with the future vision of AC, where a manager may simply specify the business objectives and the system will make it so in terms of the needed ICT [58]. A policy-based management tool may reduce the complexity of product and system management by providing uniform cross-product policy definition and management infrastructure [15].

3.9 Related initiatives

Other self-managing initiatives include HP (Adaptive Infrastructure) [59], Sun (N1) [60], Microsoft (Dynamic Systems Initiative) [61], Cisco (Adaptive Network Care) [62], Intel (Proactive Computing) [63], which all conclude that the only viable long-term solution is to create computer systems that manage themselves.

The latest related research initiative is autonomic communications [39,64,65]. A European Union brainstorming

workshop in July 2003 to discuss novel communication paradigms for 2020 identified autonomic communications as an important area for future research and development [66]. This can be interpreted as further work on self-organising networks, but is undoubtedly a reflection of the growing influence of the AC initiative. Autonomic communications has the same motivators as the AC concept, with particular focus on the communications research and development community, so as to understand how an autonomic network element's behaviours are learned, influenced or changed, and how in turn, these affect other elements, groups and networks. The ability to adapt the behaviour of the elements was considered particularly important in relation to drastic changes in the environment, such as technical developments or new economic models [66]. This initiative has now evolved into a major European research program known as Situated and Autonomic Communications (SAC) [67].

3.10 Related paradigms

Related initiatives, as in perceived future computer paradigms, include grid computing, utility computing, pervasive computing, ubiquitous computing, invisible computing, world computing, ambient intelligence, ambient networks and so on. The driving force behind these future paradigms of computing is the increasing convergence between the following technologies [68]:

- Proliferation of devices
- Wireless networking
- Mobile software

Weiser first described what has now become known as ubiquitous computing [69] as the move away from the dramatic machine (where hardware's and software's focus was on being so exciting that we as users would not want to be without it) toward making the machine invisible (so embedded in our lives it is used without thinking or recognising it as computing). Behind these different terms and research areas lie three key properties [68]:

- nomadic
- embedded
- invisible

In effect, this may lead to the creation of a single system with (potentially) billions of networked information devices. All of these next-generation paradigms, in one form or another, will require an autonomic—self-managing—infrastructure to thus provide the successful reality of this envisaged level of invisibility and mobility.

Currently, and for the foreseeable future, the majority of users access computing through personal devices. Personal computing offers unique challenges for self-management due to its multiequipment, multisituation and multiuser nature. Personal AC is much less about achieving optimum performance or exploiting redundancy (as in AC) and more about simplifying use of the equipment and the associated services [12,18]. As such, it is particularly relevant to deal with the move toward a nomadic, embedded and invisible computing future [70,71].

4 Research and technology transfer issues

The challenge of AC requires more than the re-engineering of today's systems. Autonomic computing also requires new ideas, new insights and new approaches.

Some of the key issues that will need to be addressed are

- Trust
- Economics
- Standards, standards and more standards

Trust

Even if the community manages to get the technology right, the trust of the user will be an issue in terms of handing over control to the system. AI and autonomous agent domains have suffered from this problem. For instance, neural networks (concerns over black box approach) and uncertainty in AI techniques are often not adopted. Rule-based systems often win over, even with all their disadvantages, because the user can trace and understand (and thus implicitly trust) them [38]. Note that, even within AC and autonomic communications, the majority of literature assumes rules will be used instead of other less brittle and more adaptable stochastic AI approaches.

Economics

New models of reward will need to be designed. Autonomy and autonomicity may derive another self-* property; selfishness. For instance, why would an autonomic element perform an operation, e.g. pass on information, for another AE that was outside its organisation and did not affect or benefit it? In particular, if it was operating within a mobile (battery-powered) environment and to do so incurred personal cost.

Standards

The overarching vision of AC will only be achievable through standards, in particular, for communicating between AEs. At the same time, there needs to be agile ways to define these—the self-defining property will be key here.

5 Further and recommended reading

The best starting point for further reading is IBM's call-to-arms launch of the initiative [1], the vision paper [72] and dawning paper [2] as well as news about the initiative [73].

Since the launch of AC, IBM has released various white papers, for instance [11,14,62], on their autonomic web sites

[74,75]. The general concepts within these have essentially been brought together into a book published by IBM Press [76]. Although it lacks good citations, it covers IBM's view of autonomicity and how it strategically fits within their other initiatives (such as on-demand).

Delving deeper into the autonomic research, origins of some of the IBM thinking on this can be attributed to the active middleware services (AMS) community, where their fifth workshop in Seattle in 2003 became the Autonomic Computing Workshop [77] and evolved with IBM's backing into the Autonomic Conference (New York 2004) [78]. The early focus at this stage is very much on its roots: middleware,infrastructures and architectures. Other autonomic workshops include [79–85] and related workshops, such as [86,87].

Special issue journals are beginning to appear [24,88]. The papers in [24] generally cover engineering topics, such as mirroring and replication of servers, software hot swapping and DB query optimisation, and those in [88] strongly represent autonomic efforts for the grid, web and networks. Appreciating the wider context, the boiling pot that influenced AC, can be found in other research initiatives, such as recovery-oriented computing [89].

6 Defining terms, terminology and glossary

AE	Autonomic element (consists of autonomic manager and managed component)
AM	Autonomic manager (consists of control loop and components to provide self-* for the managed component)
Autonomic communications	Same motivators as the AC concept with particular focus on the communications research and development community (see SAC)
Autonomic computing	Overarching initiative to create self-managing computer-based systems, metaphor inspired by the biological autonomic nervous system
Autonomic systems	Often used as alternative term for AC; at times used as autonomicity from systems perspective, or even that Autonomic systems = Autonomic computing + Autonomic communications
Autonomicity	Has autonomic capability
Autonomics	Used, as in, has autonomic capability (dictionaries would indicate autonomicity as the correct term but autonomics may become common place in computing)
Control loop	Closed loop of feedback control
Effectors	Defined means to bring about change to a part of the managed component (alt. actuators)
Environment awareness	Aware of current external operating conditions and knowledge of abilities. From this perspective may be considered a part of self-awareness—one's place in the environment Another view of environment awareness is that where the environment is aware of the individuals themselves, for instance, through their heartbeat or pulse
HBM	Heart-beat monitor
MAPE	Monitor, analyse, plan and execute components within autonomic manager
MC	Managed component—component that is protected by the AM
PBM	Pulse monitor (extension of HBM with health/urgency tones)
SAC	Situated and autonomic communications; situated (as in reacting locally on environment and context changes) and self-managed, i.e. communication and networking vision of being task- and knowledge-driven and fully scalable
Self-*	Self-managing properties
Self-anticipating	Ability to predict likely outcomes or simulate self-* actions
Self-awareness	Know thy self; awareness of its internal state, knowledge about past states and operating abilities
Self-chop	Initial four (and generic) self-properties (configuration, healing, optimisation and protection)
Self-configuring	Ability to configure and reconfigure to meet policies/goals
Self-critical	Ability to consider if policies are being met or goals achieved (alt. self-reflect)
Self-defining	In reference to autonomic event messages between AMs contains data and definition of that data—metadata (for instance using XML). In reference to goals/policies—defining these (from self-reflection etc.)
Self-governing	As in autonomous-responsibility for achieving goals/tasks
Self-healing	Reactive (self-fixing faults) and proactive (predicting and preventing faults)
Self-managing	Autonomous plus responsibility for wider self-* management issues
Self-optimising	Optimisation of tasks and nodes
Self-organized	Organisation of effort/nodes. Particularly used as in networks/communications
Self-protecting	Ability to protect system
Self-reflecting	Ability to consider if routine and reflex operations of self-* operations are as expected May involve self-simulation to test scenarios
Self-simulation	Ability to generate and test scenarios without affecting the live system
Selfware	Self-managing software or firmware
Sensors	Means to measure a part of the managed component (alt. probes)

7 Conclusions and discussion

The journey to achieve the overarching vision of autonomic systems has just started. The revolutionary vision will involve an evolution of innovation in systems and software engineering as well as collaboration with many other diverse fields.

Early R&D presented in this paper highlights the gaining momentum in all aspects to meet the vision. The challenges of re-engineering today's systems of systems away from the complexity quagmire toward tomorrow's pervasive and ubiquitous computation and communications will require unifying standards, new economic models and trust of the users, as well as innovations to address the hard technical issues.

It has been expressed that, in AC's initial deployment take-up, many researchers and developers have zeroed in on self-optimisation because it is perceived as easier to translate into dollars [15]. Essentially, this focus on optimisation from the four self-chop attributes may be considered as going against the grain of where technology has been leading us—to faster machines—as such fine-grained optimisation is not necessarily a major concern [15]. For AC to succeed in the longer term, the other self-* attributes must be addressed equally and in an integrated fashion.

As well as addressing complexity, AC also offers the promise of a lower total cost of ownership and a reduced maintenance burden as systems become self-managing. Achieving this vision will likely make substantial demands on legacy maintenance budgets in the short-term as autonomic function and behaviour is designed into systems.

The NASA community, with its increasing utilisation of autonomy in missions, can only benefit from a paradigm shift within computing that brings autonomicity into the mainstream.

Acknowledgements The development of this paper was supported by the Centre for Software Process Technologies (CSPT), funded by Invest NI through the Centres of Excellence Programme, under the EU Peace II initiative.

References

1. Horn P (2001) Autonomic computing: IBM perspective on the state of information technology, IBM T.J. Watson Labs, NY, 15th October 2001. Presented at AGENDA 2001, Scottsdale. Available via http://www.research.ibm.com/autonomic/
2. Ganek AG, Corbi TA (2003) The dawning of the autonomic computing era. IBM Sys J 42(1):5–18
3. Sterritt R, Hinchey M (2004) Apoptosis and self-destruct: a contribution to autonomic agents?. In: Proceedings of Third NASA-Goddard/IEEE workshop on formal approaches to agent-based systems (FAABS III), Washington 26–27 April, in Lecture notes in computer science 3228. Springer-Verlag, Berlin Heidelberg New York, pp 269–278
4. Sterritt R (2002) Towards autonomic computing: effective event management. In: Proceedings of 27th annual IEEE/NASA software engineering workshop (SEW), Maryland, 3–5 December. IEEE Computer Society, pp 40–47
5. Sterritt R, Bustard DW (2003) Autonomic computing—a means of achieving dependability? In: Proceedings of IEEE international conference on the engineering of computer based systems (ECBS'03), Huntsville, 7–11 April. IEEE CS Press, pp 247–251
6. Avižienis A, Laprie J-C, Randell B, Landwehr C (2004) Basic concepts and taxonomy of dependable and secure computing. IEEE Transactions on Dependable and Secure Computing, 1(1), Jan–Mar
7. USA Computing Research Association. (2002) Grand research challenges. Available via http://www.cra.org/reports/gc.systems.pdf
8. UK grand challenges for computing research. (2002) Available via http://www.nesc.ac.uk/esi/events/Grand_Challenges/
9. Tianfield H (2003) Multi-agent based autonomic architecture for network management, Industrial Informatics, 2003. INDIN 2003. In: Proceedings of the IEEE international conference, 21–24 August. pp 462–469
10. Sterritt R, Bustard DW (2003) Towards an autonomic computing environment. In: Proceedings of IEEE DEXA 2003 workshops—1st international workshop on autonomic computing systems, Prague, Czech Republic, 1–5 September. pp 694–698
11. IBM (2003) An architectural blueprint for autonomic computing
12. Bantz DF, Bisdikian C, Challener D, Karidis JP, Mastrianni S, Mohindra A, Shea DG, Vanover M (2003) Autonomic personal computing. IBM Sys J 42(1):165–176
13. Deen G, Lehman T, Kaufman J (2003) The almaden optimalgrid project. In: IEEE Proceedings of autonomic computing workshop, 5th AMS, Seattle. pp 14–21
14. IBM (2001) Autonomic computing concepts. White Paper, IBM
15. Ganek AG (2003) Autonomic computing: implementing the vision. Keynote presentation at the autonomic computing workshop, AMS 2003, Seattle, 25th June
16. Sterritt R (2003) Pulse monitoring: extending the health-check for the autonomic GRID. In: Proceedings of IEEE workshop on autonomic computing principles and architectures (AUCOPA 2003) at INDIN 2003, Banff, Alberta, Canada, 22–23 August. pp 433–440
17. Sterritt R, Gunning D, Meban A, Henning P (2004) Exploring autonomic options in an unified fault management architecture through reflex reactions via pulse monitoring. In: Proceedings of IEEE workshop on the engineering of autonomic systems (EASe 2004) at the 11th annual IEEE international conference and workshop on the engineering of computer based systems (ECBS 2004), Brno, Czech Republic, 24–27 May. pp 449–455
18. Bantz DF, Frank D (2003) Challenges in autonomic personal computing, with some new results in automatic configuration management. In: Proceedings of IEEE workshop on autonomic computing principles and architectures (AUCOPA 2003) at INDIN 2003, Banff, Alberta, Canada, 22–23 August. pp 451–456
19. Brown AB, Hellerstein J, Hogstrom M, Lau T, Lightstone S, Shum P, Peterson Yost M (2004) Benchmarking autonomic capabilities: promises and pitfalls, international conference on autonomic computing (ICAC'04). pp 266–267
20. Lightstone S (2003) Towards benchmarking—autonomic computing maturity invited talk at workshop on autonomic computing principles and architectures (AUCOPA' 2003) at the IEEE international conference industrial informatics (INDIN 2003), Banff, Alberta, Canada, 21–24 August
21. Kistler-Glendon K (2004) Beginning the autonomic journey—a review and lessons learned from an autonomic computing readiness assessment at a major US telecom, CHIACS2 Conference
22. IBM. Alpha works autonomic computing site. Available via http://www.alphaworks.ibm.com/autonomic
23. Sterritt R (2003) Autonomic computing: the natural fusion of soft computing and hard computing. In: Proceedings of 2003 IEEE international conference on systems, man and cybernetics, Washington, 5–8 October. pp 4754–4759
24. IBM systems journal (2003) special issue on autonomic computing, Vol. 42(1), pp 197, available via http://www.research.ibm.com/journal/sj/421/
25. Norman DA, Ortony A, Russell DM (2003) Affect and machine design: lessons for the development of autonomous machines. IBM Sys J 42(1):38–44
26. Sloman A, Croucher M (1981) Why robots will have emotions. In: Proceedings of 7th international joint conference on AI, Vancouver. pp 197–202

27. Sloman A (1999) Review of: Rosalind Picard's affective computing, MIT Press, 1997. AI Magazine
28. Russell LW, Morgan SP, Chron EG (2003) Clockwork: a new movement in autonomic systems. IBM Sys J 42(1):77–84
29. Russell LW, Morgan SP, Chron EG (2003) On-line model selection procedures in clockwork, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
30. Guo H (2003) A Bayesian approach for autonomic algorithm selection, IJCAI Workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
31. Aiber S, Etzion O, Wasserkrug S (2003) The utilization of AI techniques in the autonomic monitoring and optimization of business objectives, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
32. Trumler W, Bagci F, Petzold J, Ungerer T (2003) Smart doorplates–toward an autonomic computing system. In: Autonomic computing workshop fifth annual international workshop on active middleware services (AMS'03), pp 42
33. Trumler W, Petzold J, Bagci F, Ungerer T (2004) AMUN— autonomic middleware for ubiquitous environments applied to the smart doorplate project, international conference on autonomic computing (ICAC'04), pp 274–275
34. Lau T, Oblinger D, Bergman L, Castelli V, Anderson C (2003) Learning procedures for autonomic computing, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
35. IBM Tivoli Monitoring, IBM Corp. Available at http://www.tivoli.com/products/index/monitor/
36. Lanfranchi G, Della Peruta P, Perrone A, Calvanese D (2003) Toward a new landscape of systems management in an autonomic computing environment. IBM Sys J 42(1):119–128
37. Sterritt R, Bustard DW, McCrea A (2003) Autonomic computing correlation for fault management system evolution. In: Proceedings of IEEE international conference industrial informatics (INDIN 2003), Banff, Alberta, Canada, 21–24 August. pp 240–247
38. Sterritt R, Bustard DW (2002) Fusing hard and soft computing for fault management in telecommunications systems IEEE Trans Systems Man and Cybernetics part C, 32(2)
39. Sterritt R (2004) Autonomic networks: engineering the self-healing property, engineering applications of artificial intelligence, Vol. 17, No. 7, Elsevier, ISSN 0952–1976. pp 727–739
40. Sahoo RK, Rish I, Oliner AJ, Gupta M, Moreira JE, Ma S, Vilalta R, Sivasubramaniam A (2003) Autonomic computing features for large-scale server management and control, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
41. Littman ML, Nguyen T, Hirsh H (2003) A model of cost-sensitive fault mediation, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
42. Cheliotis G, Kenyon C (2003) Autonomic economics: a blueprint for self-managed systems, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
43. Parkes DC (2003) Five AI challenges in strategy proof computing, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
44. Kaiser G, Parekh J, Gross P, Valetto G (2003) Kinesthetics extreme: an external infrastructure for monitoring distributed legacy systems. In: Proceedings of the autonomic computing workshop, 5th international workshop on active middleware services (AMS 2003), Seattle. pp 22–30
45. Birman KP, van Renesse R, Vogels W (2003) Navigating in the storm: using astrolabe for distributed self-configuration, monitoring and adaptation. In: Proceedings of the autonomic computing workshop, 5th international workshop on active middleware services (AMS 2003), Seattle. pp 4–13
46. Agarwal M, Bhat V, Liu H, Matossian V, Putty V, Schmidt C, Ahang G, Zhen L, Parashar M, Khargharia B, Hariri S (2003) Automate: enabling autonomic applications on the grid. In: Proceedings of the autonomic computing workshop, 5th international workshop on active middleware services (AMS 2003), Seattle. pp 22–30
47. Wyatt J, Sherwood R, Sue M, Szijjarto J (1999) Flight validation of on-demand operations: the deep space one beacon monitor operations experiment, 5th international symposium on artificial intelligence, robotics and automation in space (i-SAIRAS '99), ESTEC, Noordwijk, The Netherlands. 1–3 June
48. Truszkowski W, Hinchey M, Rash J, Rouff C (2004) NASA's swarm missions: the challenge of building autonomous software. IEEE IT Professional mag., September/October. pp 51–56
49. Clancy DJ (2002) NASA challenges in autonomic computing, Almaden Institute 2002, IBM Almaden Research Center, San Jose, 10th April
50. Hughes PM (2003) Application of autonomic computing concepts for GSFC's next generation missions, presentation at the Woodrow Wilson International Center for Scholars, 28th October
51. Sterritt R (2004) SelfWares-episode IV–autonomic computing: a new hope, NASA GSFC IS &T Colloquium, 1 December
52. Truszkowski W, Rash J, Rouff C, Hinchey M (2004) Asteroid exploration with autonomic systems. In: Proceedings of IEEE workshop on the engineering of autonomic systems (EASe 2004) at the 11th annual IEEE international conference and workshop on the engineering of computer based systems (ECBS 2004), Brno, Czech Republic, 24–27 May. pp 484–490
53. Padget J (2003) The role of norms in autonomic organizations, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
54. Therani M, Zeng D, Dror M (2003) Decentralized resource management in autonomic systems, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
55. Gutierrez RLZ, Huhns MN (2003) Achieving software robustness via multiagent-based redundancy, IJCAI workshop on AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August
56. Jennings NR, Wooldridge M (2000) Agent-oriented software engineering. In: Bradshaw J (ed) Handbook of agent technology. AAAI/MIT Press, Cambridge, Massachusetts
57. Huhns MN, Holderfield VT, Gutierrez RLZ (2003) Robust software via agent-based redundancy. In: Second international joint conference on autonomous agents and multiagent systems, AAMAS 2003, 14–18 July, Melbourne. pp 1018–1019
58. Lymberopoulos L, Lupu E, Sloman M (2003) An adaptive policy-based framework for network services management. J Netw Syst Manage 11(3)
59. HP World (2003) Adaptive Infrastructure, Atlanta Georgia, 11–15 August
60. Sun Microsystems (2002) N1—Introducing just-in-time computing, White paper
61. Microsoft Corporation (2004) Dynamic systems initiative overview, white paper, 31 March 2004, revised 15 November
62. IBM and Cisco Systems (2003) Adaptive services framework, white paper, version 1.0, October 14
63. Want R, Pering T, Tennenhouse D (2003) Comparing autonomic and proactive computing. IBM Sys J 42(1):129–135
64. Sterritt R (2003) xACT: autonomic computing and telecommunications. BT Exact Research Fellowship
65. Autonomic Communications (2004) Available via http://www.autonomic-communication.org/
66. EU IST FET (2003) New communication paradigms for 2020, brain storming meeting, Brussels, Belgium, (report published Sept 2003)
67. EU IST FET (2004) Situated and autonomic communications (COMS)— communication paradigms for 2020 proactive initiative 2004
68. The future of computing project (2004) Available via http://www.thefurtureofcomputing.org

69. Weiser M (1994) Creating the invisible interface. Symposium on user interface software and technology. ACM Press, New York
70. Sterritt R, Bantz DF (2004) PAC-MEN: personal autonomic computing monitoring environments. In: Proceedings of IEEE DEXA 2004 workshops—2nd international workshop on self-adaptive and autonomic computing systems (SAACS 04), Zaragoza, Spain, August 30th–September 3rd, IEEE. pp 737–741
71. Sterritt R, Chung S (2004) Personal autonomic computing self-healing tool. In: Proceedings of IEEE workshop on the engineering of autonomic systems (EASe 2004) at 11th annual IEEE international conference and workshop on the engineering of computer based systems (ECBS 2004), Brno, Czech Republic, 24–27 May. pp 513–520
72. Kephart JO, Chess DM (2003) The vision of autonomic computing. IEEE Comp, pp 41–50
73. Paulson LD (2002) Computer system, Heal Thyself. IEEE Comp, pp 20–22
74. IBM Autonomic Website. Available via http://www.ibm.com/autonomic/
75. IBM Autonomic Research Website. Available via http://www.research.ibm.com/autonomic/
76. Murch R (2004) Autonomic computing. IBM Press and Prentice Hall, Englewood Cliffs, New Jersey, ISBN 0-13-144025-X
77. The autonomic computing workshop, 5th international workshop on active middleware services (AMS 2003), Seattle. In: Proceedings IEEE Computer Society, 25th June 2003. p 198
78. IEEE international conference on autonomic computing (ICAC'04). New York, 17–18 May 2004
79. Almaden institute symposium: autonomic computing, Almaden, April 2002
80. IJCAI Workshop, AI and autonomic computing: developing a research agenda for self-managing computer systems, Acapulco, Mexico, 10th August. Available via http://www.research.ibm.com/ACworkshop
81. Workshop on autonomic computing principles and architectures (AUCOPA' 2003), at INDIN 2003— first IEEE conference on industrial informatics, Banff, Canada
82. First international workshop on autonomic computing systems at 14th international conference on database and expert systems applications (DEXA'2003), Prague, Czech Republic, 1–5 September, 2003
83. Autonomic applications workshop, at international conference high performance computing (HiPC 2003), Taj Krishna, Hyberabad, India, 17th December
84. IEEE workshop on the engineering of autonomic systems (EASe 2004) at the 11th annual IEEE international conference and workshop on the engineering of computer based systems (ECBS 2004), Brno, Czech Republic, 24–27 May 2004
85. Second IEEE Workshop on the engineering of autonomic systems (EASe 2005) at the 12th annual IEEE international conference and workshop on the engineering of computer based systems (ECBS 2005) and 29th NASA/IEEE software engineering workshop (SEW), Washington, 3–8 April 2005
86. Workshop on self-healing, adaptive and self-MANaged systems(SHAMAN), New York, 23 June, 2002
87. Workshop on self-healing systems (WOSS 2002), Charleston, November 2002
88. Special issue on autonomic computing systems, engineering applications of artificial intelligence, vol 17, no, 7. Elsevier, Berlin Heidelberg New York, ISSN 0952-1976, October 2004. pp 689–869
89. Brown A, Patterson DA (2001) Embracing failure: a case for recovery-oriented computing (ROC), 2001 high performance transaction processing symposium. Asilomar